We are IntechOpen, the world's leading publisher of Open Access books
Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS
BOOK CITATION INDEX
INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

Chapter

Fast Motion Estimation's Configuration Using Diamond Pattern and ECU, CFM, and ESD Modes for Reducing HEVC Computational Complexity

Randa Khemiri, Nejmeddine Bahri, Fatma Belghith, Soulef Bouaafia, Fatma Elzahra Sayadi, Mohamed Atri and Nouri Masmoudi

Abstract

The high performance of the high efficiency video coding (HEVC) video standard makes it more suitable for high-definition resolutions. Nevertheless, this encoding performance is coupled with a tremendous encoding complexity compared to the earlier H264 video codec. The HEVC complexity is mainly a return to the motion estimation (ME) module that represents the important part of encoding time which makes several researches turn around the optimization of this module. Some works are interested in hardware solutions exploiting the parallel processing of FPGA, GPU, or other multicore architectures, and other works are focused on software optimizations by inducing fast mode decision algorithms. In this context, this article proposes a fast HEVC encoder configuration to speed up the encoding process. The fast configuration uses different options such as the early skip detection (ESD), the early CU termination (ECU), and the coded block flag (CBF) fast method (CFM) modes. Regarding the algorithm of ME, the diamond search (DS) is used in the encoding process through several video resolutions. A time saving around 46.75% is obtained with an acceptable distortion in terms of video quality and bitrate compared to the reference test model HM.16.2. Our contribution is compared to other works for better evaluation.

Keywords: HEVC, motion estimation, early skip detection (ESD), early CU termination (ECU), coded block flag (CBF) fast method (CFM)

1. Introduction

The fast multimedia technology development and network communications makes ultrahigh-definition (HD) and HD video contents widely used in our daily life. This fast jump to use high video resolutions in which many provide some problems in terms of memory storage cost and transmission bandwidth gives birth

to the new high efficiency video coding (HEVC) [1, 2]. HEVC is developed in 2013 by the joint collaborative team on video coding (ISO/IEC) Moving Picture Experts Group (MPEG) and the International ITU-T Video Coding Experts Group (VCEG). It is urbanized to overcome the enormous amount of UHD video contents. Compared to the earlier H.264/AVC [3] standard and at the identical visual quality, HEVC guarantees a high encoding performance, reaching 50% of bitrate [4]. Facing to this immense huge encoding performance, a huge computational complexity is obtained. Motion estimation (ME) represents the large part of encoding process that occupies around 70% of the total time of inter prediction, as Jungho [5] indicates in **Figure 1**.

This large consumption is principally due to the new hierarchy of the block coding based on coding tree units (CTU). This new concept is analog to macroblocks in the earlier standard of compression. Each picture frame is divided into square forms, called coding units (CUs) [6], where 64 × 64 represents the maximum size, and recursively subdivided into 8 × 8 blocks. Prediction and

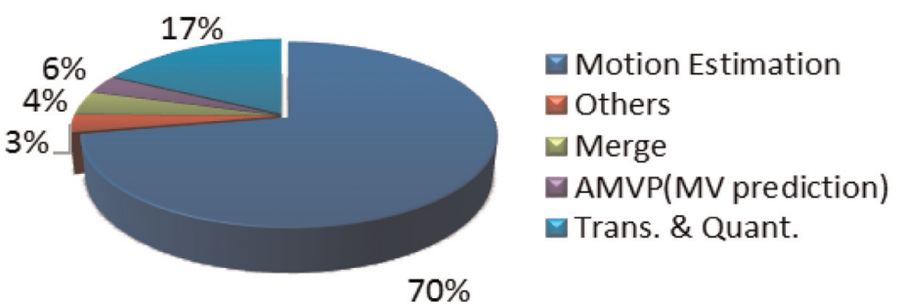

Figure 1.
Encoding time distribution.

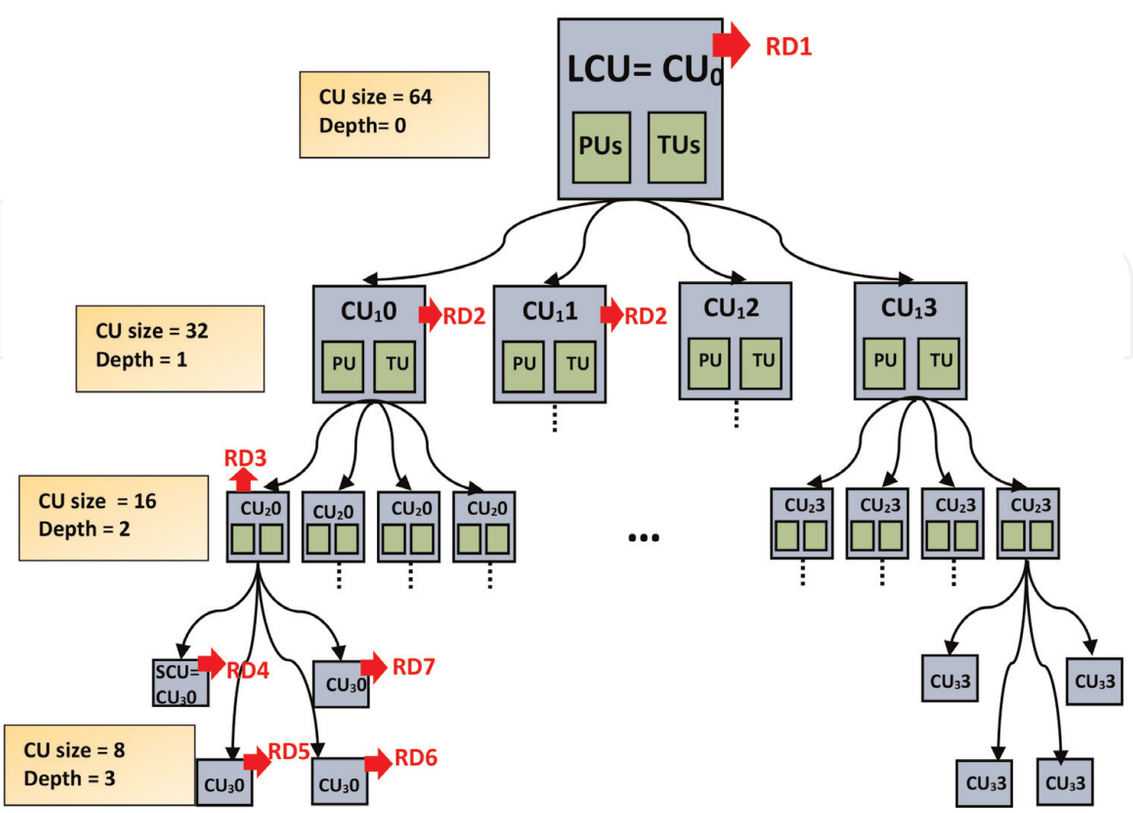

Figure 2.
CTU tree structure in the HEVC standard.

transform blocks (PUs and TUs) are in each CU, where PU represents the principal unit in the ME process.

Figure 2 shows the CTU tree structure in the HEVC standard where LCU represents the large coding unit and SCU represents the small coding unit.

When reducing the time essential for the search algorithm, the ME computational complexity will be automatically reduced. Furthermore, when using different fast mode decision algorithms based on early termination, the ME computational complexity will be reduced, which primes to the entire HEVC execution time reduction.

It is within this context that this article presents a fast encoding algorithm principally based on the early skip detection (ESD), the coded block flag (CBF) fast method (CFM), and the early CU termination (ECU) modes [7–9] to decrease the HEVC encoding complexity.

The remainder of this paper is structured as follows: the next section details some works on the HEVC fast motion estimation algorithms. Section 3 provided an overview of the motion estimation algorithm. Section 4 highlights the proposed fast configuration for the HEVC encoder. Experimental results for the fast HEVC configuration compared to the results obtained with the original HM16.2 reference software [10] are discussed in Section 5. Finally, in Section 6, conclusions and some prospects are given.

2. Related works

Aiming to optimize the HEVC encoder complexity, several works have been proposed to reduce the test zonal search (TZS) motion estimation algorithm. Some works are interested in hardware solutions, and others are focused on software optimizations.

In [11], using sequential and parallel techniques, two hardware diamond architectures for HEVC video coding are proposed. These architectures achieve an encoding in full HD at 30 frames per second using a Virtex-7 field programmable gate array (FPGA) design.

Authors in [12] have proposed a hardware parallel sum of absolute difference (SAD) design for gray-scale images to reduce motion estimation time for block size of 4×4 pixels. A multiplier is exploited for addition as a partial product reduction (PPR). Results obtained on Virtex-2 Xilinx FPGA show that the maximum frequency obtained is 133.2 MHz for 4×4 block size. Nalluri et al., in [13, 14], have proposed two other SAD architectures on FPGA Xilinx Virtex without and with parallelism. The proposed parallel architecture has accelerated the SAD calculation by 3.9× compared to the serial SAD architecture. In [15], authors have proposed two implementations of the SAD and SSD algorithms using NVIDIA GeForce GTX480 with CUDA language in order to reduce the ME run-time. The proposed architecture saved about 32% of encoding time for class E video sequences with nonsignificant degradation in the PSNR and the bitrate.

Regarding software solutions, the 8-point square and the 8-point diamond have been replaced by Nalluri et al. [16] with a 6-point hexagonal in the TZS ME algorithm, and 50% in encoding time is saved without degradation in bitrate and PSNR. To replace the TZS algorithm, in [17, 18], authors proposed small diamond pattern search (SDPS), large diamond pattern search (LDPS), and horizontal diamond search (HDS). Experiments using HM8.0 showed that these algorithms allow a reduction of 49% of motion estimation calculation time with nonsignificant increase in bitrate and slight degradation in video quality.

In [19], Liquan et al. have proposed a fast mode decision algorithm by skipping some depths. The proposed work allows saving about 21.5% of encoding time with a slight bitrate increase and a negligible efficiency loss coding. The algorithm proposed by Qin [20] uses the ECU algorithm according to an adaptive MSE threshold value. This work ensures time saving without degradation in the quality. Podder [21] has also proposed an interesting software method to reduce the ME time. Based on human visual features (HVF), an efficient decision of the appropriate block partitioning mode has been obtained. This work allows saving 41.44% of the execution time for SCVS video sequences. In the work published in [22], a fast HEVC ME based on DS and three fast mode decisions, ECU, ESD, and CBF modes, have been presented. Simulation results show a reduction of 56.75% in the complexity of HEVC in terms of execution time, accompanied with slight degradation in video quality and bitrate, when comparing the HM.16.2 executed on an Intel® Core TM i7–3770 @3.4 GHz processor. Authors in [22] have tested just one sequence from each class with just two quantification parameters (QPs), QP = 22 and 37, to evaluate the use of the fast modes.

By analyzing all these previous works, we can note that using fast mode decision algorithms represents an interesting technique in order to reduce the HEVC computational complexity.

3. Overview of the motion estimation in the new HEVC

TZ Search algorithm, used in HEVC ME process (**Figure 3**), includes four distinct main stages in order to determine the best motion vector.

These stages, which are the motion vector prediction (MVP), the first search performed with a pattern of square or diamond forms, the refinement, and the raster search, are described in the next subsections.

3.1 Motion vector prediction (MVP)

To compute the corresponding block's median predictor, the TZS algorithm uses the up predictor, the upper right predictor, and the left predictor (**Figure 4**).

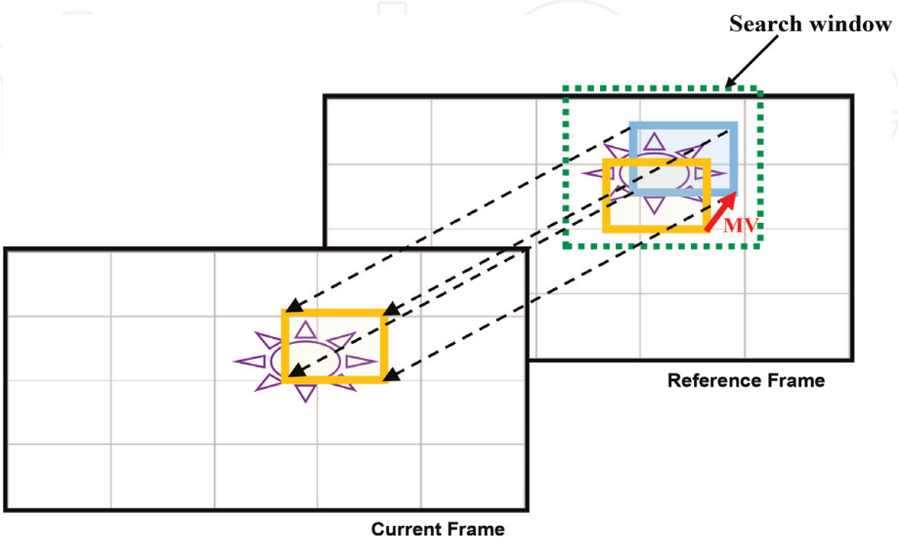

Figure 3.
Motion estimation process.

The median computation is done via the following equation.

$$Median(A, B, C) = A + B + C - \text{Min}(A, \text{Min}(B, C)) - \text{Max}(A, \text{Max}(B, C)) \quad (1)$$

3.2 Initial grid search

The first search is performed by the determination of the search pattern and the "searchrange." As it is detailed in **Figure 5(a)** and **(b)**, the main goal of this stage is to localize the search window via a pattern of square or a diamond forms.

Thus, these two search patterns are referred to the eight points for each round. The distance corresponding to the minimum distortion point is saved in the "BestDistance" variable. Currently, diamond search pattern is used as default, but the square pattern search can also be used by modifying the HEVC configuration file through the "Diamondsearch" variable.

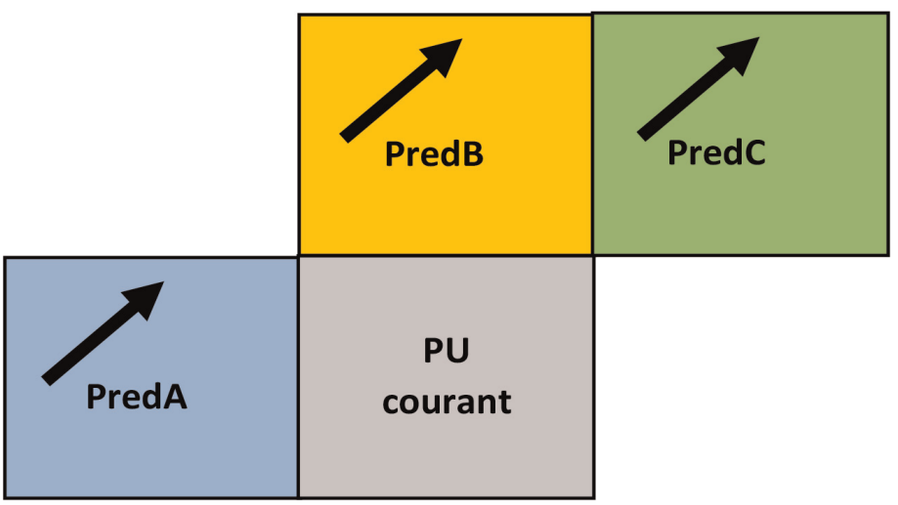

Figure 4.
MV adjacent of a current PU.

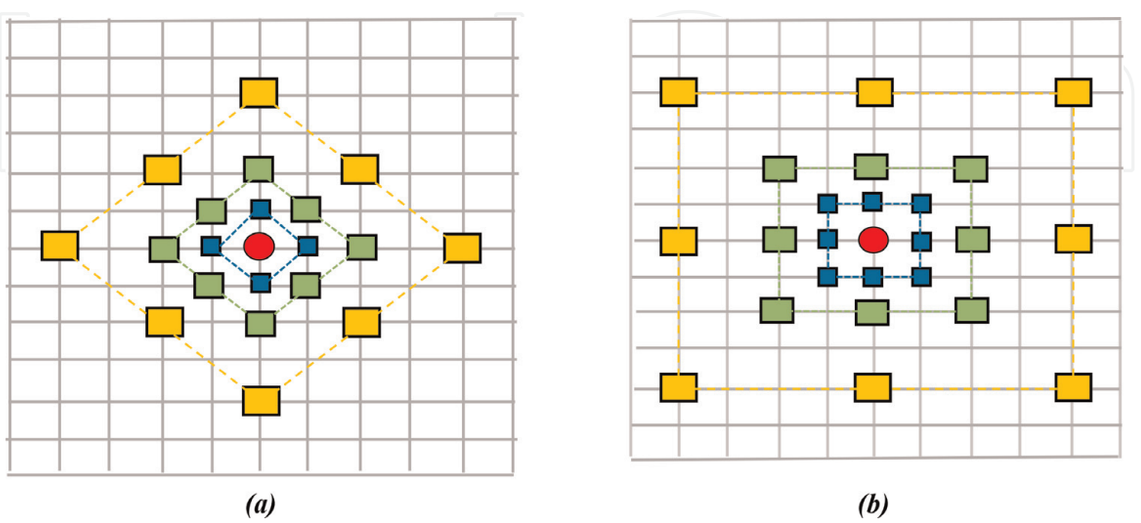

Figure 5.
Diamond/square search pattern. (a) Diamond search pattern stride length equal to 4. (b) Square search pattern stride length equal to 4.

3.3 Raster search

This step consists of choosing the distance which corresponds to the greatest matched point from the previous search. Three cases according to this distance denoted as "BestDistance" are summarized as follows:

- The process is stopped when "BestDistance" = 0.

- A refinement is needed when 1 < "BestDistance" < iRaster.

In the configuration file, "iRaster" represents a changeable variable not to be overdone.

- BestDistance > iRaster is agreed correctly; a raster scan is achieved using the iRaster value as the length step. If difference obtained from the starting station to the MV from the first level is besides large, this step is preceded. This step is computed on the entire search window.

Figure 6 shows an example of a full search algorithm with iRaster which is equal to 4.

3.4 Raster and star refinement

The refinement is performed when the distance of the motion vector previously obtained is different to 0. There are mainly two refinement types:

- Raster refinement

The best point obtained from the previous steps corresponds to the start point of the star refinement. It can be performed using a diamond or a square pattern with distances ranging from "search range" to one. In each iteration, the distance is divided by 2, and when the distance will be equal to one, two adjacent point searches are performed, and then the process is stopped.

- Star refinement

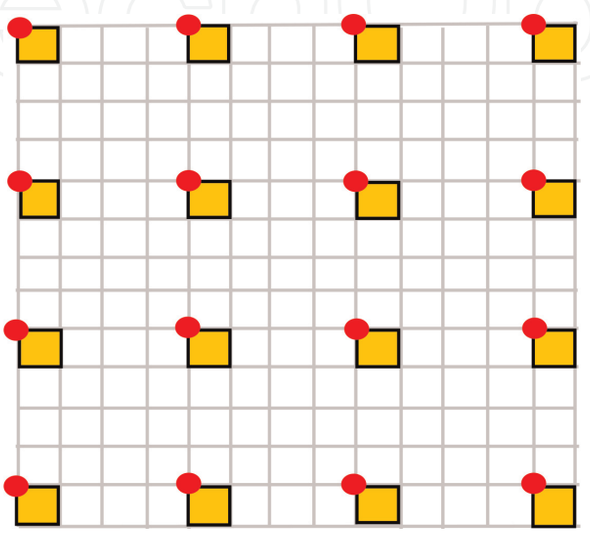

Figure 6.
Raster search pattern when iRaster = 4.

In this step, the selected point obtained from the previous steps corresponds to the start point of the star refinement. In each iteration, the distance is divided by two, and when the distance will be equal to one, two adjacent point searches are

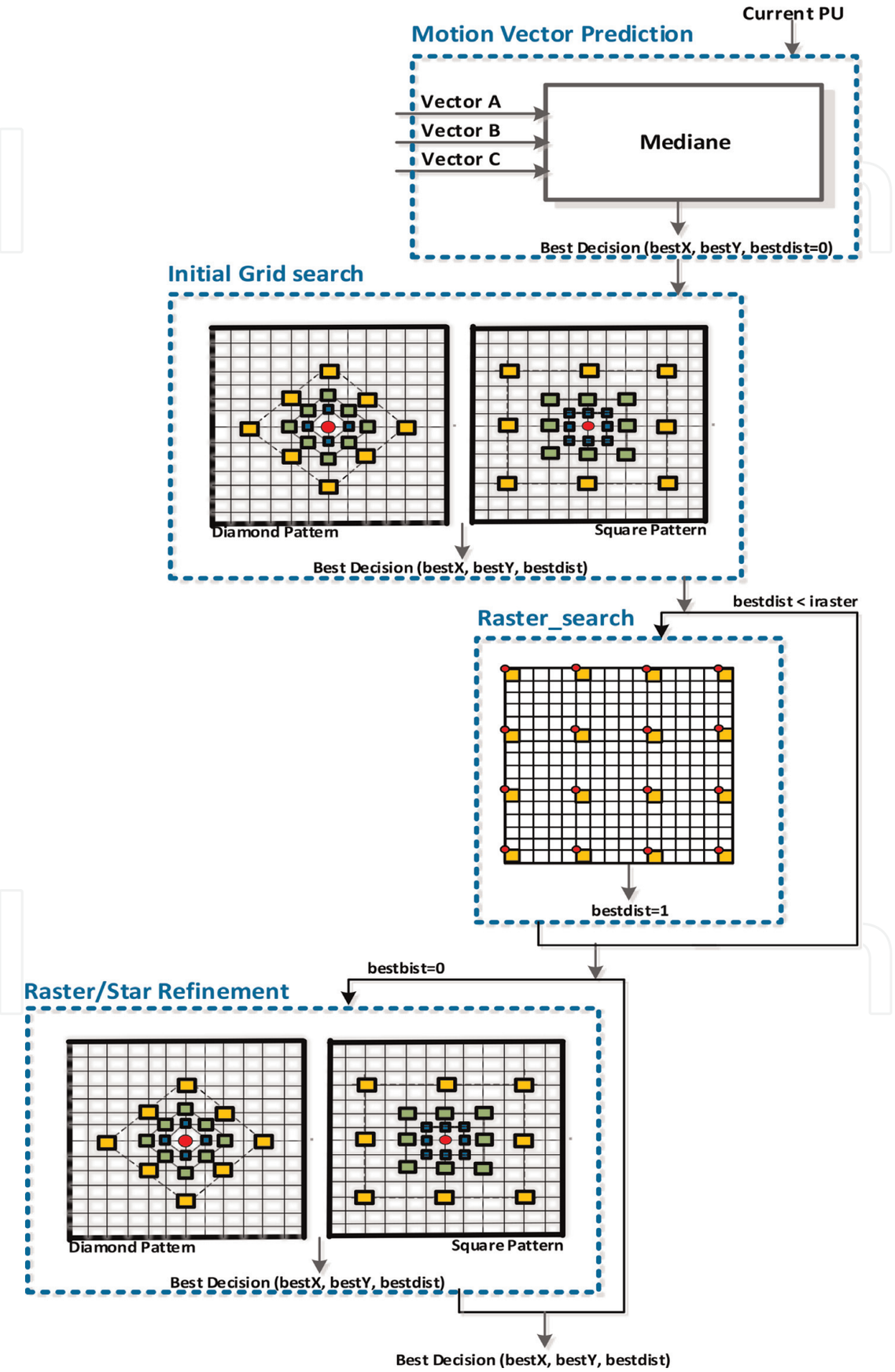

Figure 7.
The used TZSearch algorithm.

applied to determine the best estimated MV which gives the minimum of SAD (**Figure 7**).

4. The proposed fast configuration

Several fast decision mode algorithms are in this effort aiming to speed up the ME process. Firstly, diamond search pattern is utilized to decrease the encoder computational complexity. Some configurations are also set, such as the early CU termination (ECU), the early skip detection (ESD), and the coded block flag (CBF) in which fast decision mode algorithms are adopted in HEVC video coding. These proposed fast algorithms were given bellow.

4.1 Early CU termination (ECU)

This algorithm is used when switching from a depth p to the next p + 1. As **Figure 8** showed, if skip is the best current CU prediction mode, the sub-tree calculations can be skipped [23]. Thus, good mode is determined with rate distortion (RD) calculation cost [24]. The minimal RD cost relates to the skip mode that caused the stop of the partitioning [25].

Several works show that the most chosen mode was the skip [25]. This clarifies the detail that an excessive enhancement is obtained when the skip mode recognition is anticipated. This mode induces a better encoder performance since it denotes a block code deprived of residual information.

4.2 Early skip detection (ESD)

The early skip detection signifies a modest verification of the two-variance motion skip conditions (CBF and differential motion vector (DMV)). As shown in **Figure 9**, this verification is performed after determining the best inter 2 N × 2 N. Before checking the skip mode, the current CU performs two inter 2 N × 2 N modes

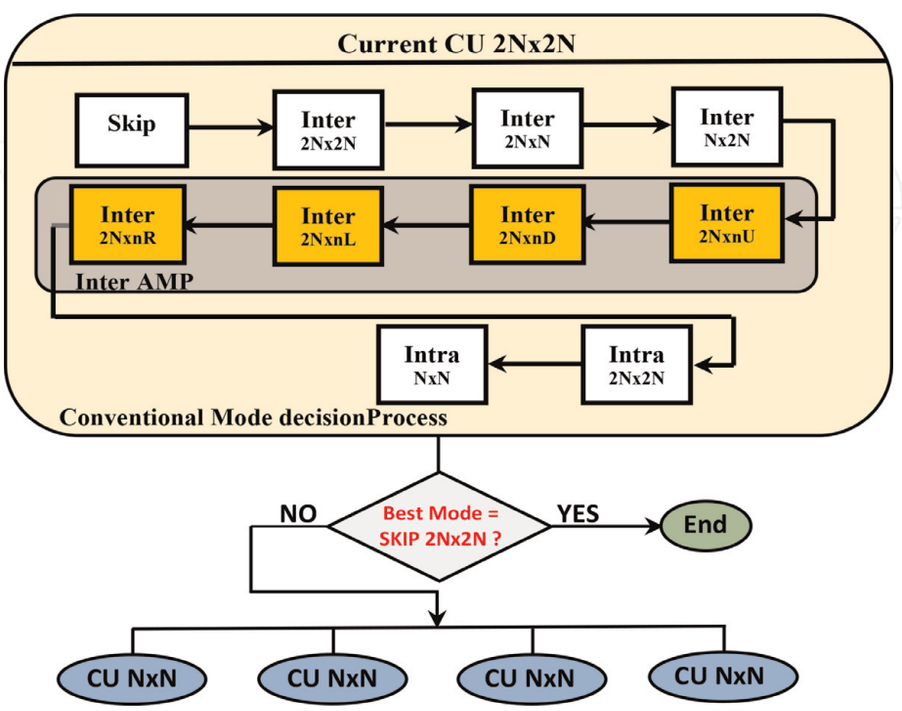

Figure 8.
Algorithm of early CU termination.

(advanced motion vector prediction called AMVP and merge mode). The DMV and CBF are checked when the minimum RD cost is induced by the mode selection. When CBF is equivalent to zero and the best mode inter 2 N × 2 N DMV is equal to (0, 0), the skip mode is the best mode of current CU. Consequently, the residual modes of PU are not examined anymore [8].

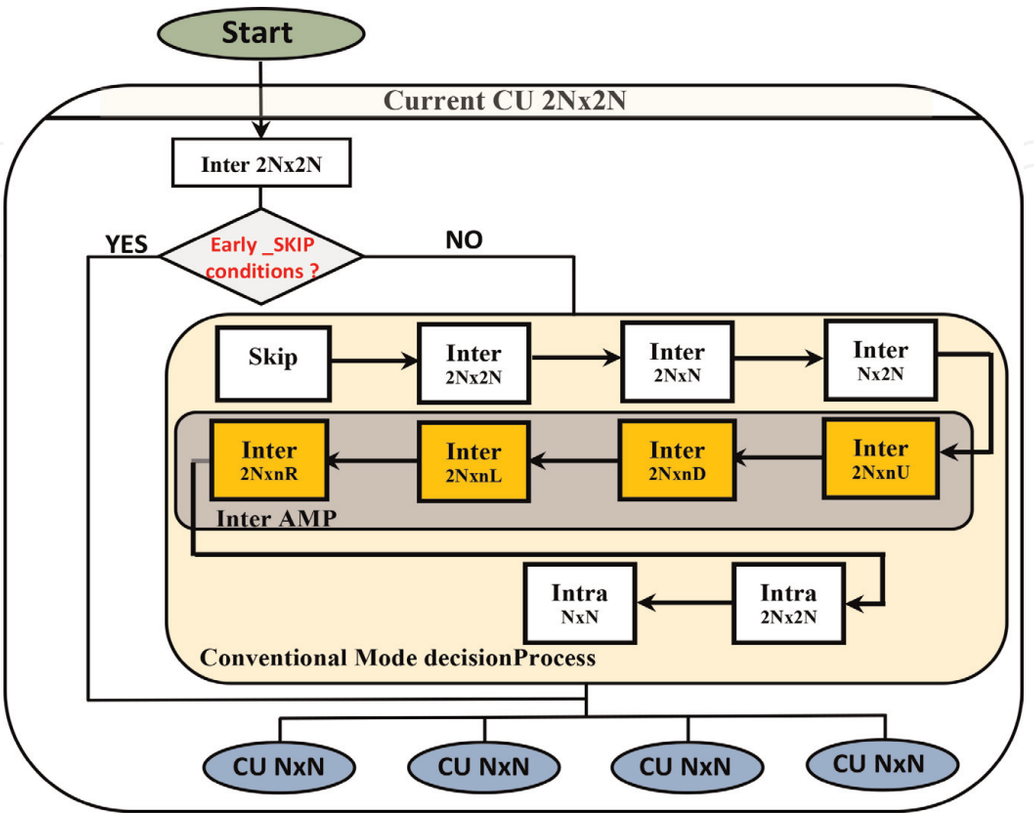

Figure 9.
Early skip detection algorithm.

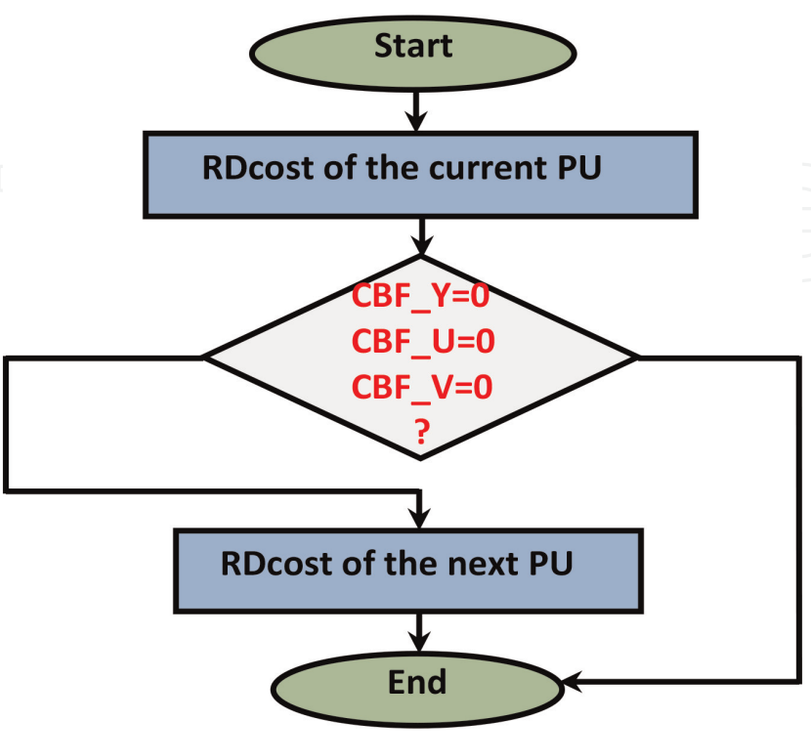

Figure 10.
Algorithm of coded block flag fast method.

4.3 Coded fast method (CFM)

The coded fast method (CFM) detects the best mode of a prediction unit [7]. As shown in **Figure 10**, for each PU mode belonging to a CU, the RD cost is calculated.

An evaluation of the different coefficients, CBF for the luminance and the two chrominances, is performed. When all transform coefficients (CBF_Y, CBF_U, and CBF_V) are equal to zero [9], all remaining modes will not be tested.

5. Experimental results

5.1 Experimental conditions

The performance evaluation of this work is effectuated with a random access (RA) configuration through the HM 16.2 reference test model, exploiting the fast mode decision algorithms ECU, ESD, and CBF, previously detailed. To appraise the fast implementation, a comparison of HEVC encoding time, bitrate, and PSNR with the original is effectuated, where a search range is 64. Sixty-four also is the CU maximal size and CU partition depth maximal equals four. An Intel® Core TM i7–3770 @ 3.4 GHz is used in this work with Windows 8 OS platform.

The four resolutions tested are to the four classes (class D (416 × 240), class C (832 × 480), class B (1920 × 1080), and class A (2560 × 1600)) [26]. For each video sequence, 50 is the encoded frame number used. To evaluate results, eight sequences recommended by the JCT-VC [26], each one with four quantification parameters (QP) 22, 27, 32, and 37, are used.

5.2 Evaluation criteria

To evaluate this work, we used the formula detailed in **Table 1**.

5.3 Results

Table 2 specifies the results obtained when using the proposed fast HEVC configuration compared to the original one.

The proposed algorithm shows a time saving of up to 46.759% on average compared to the original algorithm. The speedup attains 84.51% of encoding time for BQTerrace video for QP 37. In fact, the time saving is more important for some videos such as Traffic, BQSquare, and BQTerrace sequences ranging from 57.91 to

Criteria	Description	Formula
ΔT (%)	Encoding time speedup	$\Delta T(\%) = \frac{T_{Proposed} - T_{Original}}{T_{Original}} \times 100\ (\%)$
ΔPSNR (dB)	PSNR loss	$\Delta PSNR = \mathrm{PSNR}_{Proposed} - \mathrm{PSNR}_{Original}\ (\mathrm{dB})$
ΔBR (%)	Bitrate increase	$\Delta BR(\%) = \frac{BitRate_{Proposed} - BitRate_{Original}}{BitRate_{Original}} \times 100(\%)$

Where: $Bitrate_{original}$, $PSNR_{original}$ and $T_{original}$ represent bitrate, video quality, and encoding time of the original algorithm, respectively and $Bitrate_{proposed}$, $PSNR_{proposed}$ and $T_{proposed}$, $BitRate_{Proposed}$ represent bitrate, video quality, and encoding time of the proposed algorithm, respectively.

Table 1.
Evaluation criteria.

Class	Sequences	QP	ΔT(%)	ΔPSNR (dB)	ΔBR(%)
Class A 2560 × 1600	PeopleOnStreet	22	−25.04	−0.040	−0.470
		27	−28.37	−0.100	−0.006
		32	−39.44	−0.187	−1.086
		37	−49.69	−0.030	−2.182
	Traffic	22	−47.27	−0.103	−0.018
		27	−62.04	−0.128	−0.017
		32	−71.83	−0.180	−0.027
		37	−79.40	−0.194	−0.024
Average class A			**−50.385**	**−0.123**	**−0.478**
Class B 1920 × 1080	BQTerrace	22	−29.91	−0.060	−1.180
		27	−60.27	−0.086	−0.022
		32	−76.67	−0.081	−0.021
		37	−84.51	−0.067	−2.000
	BasketballDrive	22	−32.15	−0.015	−0.004
		27	−45.23	−0.026	−0.157
		32	−54.85	−0.059	−0.640
		37	−63.82	−0.084	−0.009
Average class B			**−55.926**	**−0.059**	**−0.504**
Class C 832 × 480	RaceHorses	22	−11.45	−0.029	−1.430
		27	−22.79	−0.073	−0.005
		32	−35.24	−0.166	−0.010
		37	−44.10	−0.230	−2.138
	PartyScene	22	−19.90	−0.040	−0.002
		27	−35.13	−0.115	−0.009
		32	−47.93	−0.168	−0.016
		37	−58.59	−0.194	−0.027
Average class C			**−34.400**	**−0.126**	**−0.454**
Class D 416 × 240	BQSquare	22	−34.64	−0.075	−0.870
		27	−54.78	−0.137	−0.016
		32	−66.90	−0.156	−0.012
		37	−75.35	−0.137	−0.810
	BlowingBubbles	22	−17.81	−0.070	−0.005
		27	−28.62	−0.110	−0.005
		32	−40.05	−0.134	−0.015
		37	−52.46	−0.115	−0.012
Average class D			**−46.326**	**−0.116**	**−0.218**
Average			**−46.759**	**−0.106**	**−0.416**

Table 2.
Performance evaluation of the proposed algorithm compared to the original one.

65.135%. This is due to the motion slowness in these sequences. Indeed, for videos containing low motion activities [18], the improvement is more significant. With the highest resolution, traffic video is characterized by intensive movement of objects against a stationary background. Concerning BQSquare, this video having fast motion is often coded by the bi-predictive mode, as it is the best prediction mode.

Defiantly for sequences with high activity, such as BlowingBubbles, RaceHorses, and PeopleOnStreet, the time saving is only around 34.73 and 28.38%. The worst case is for the motion-filled and dynamic RaceHorses video, which records horse

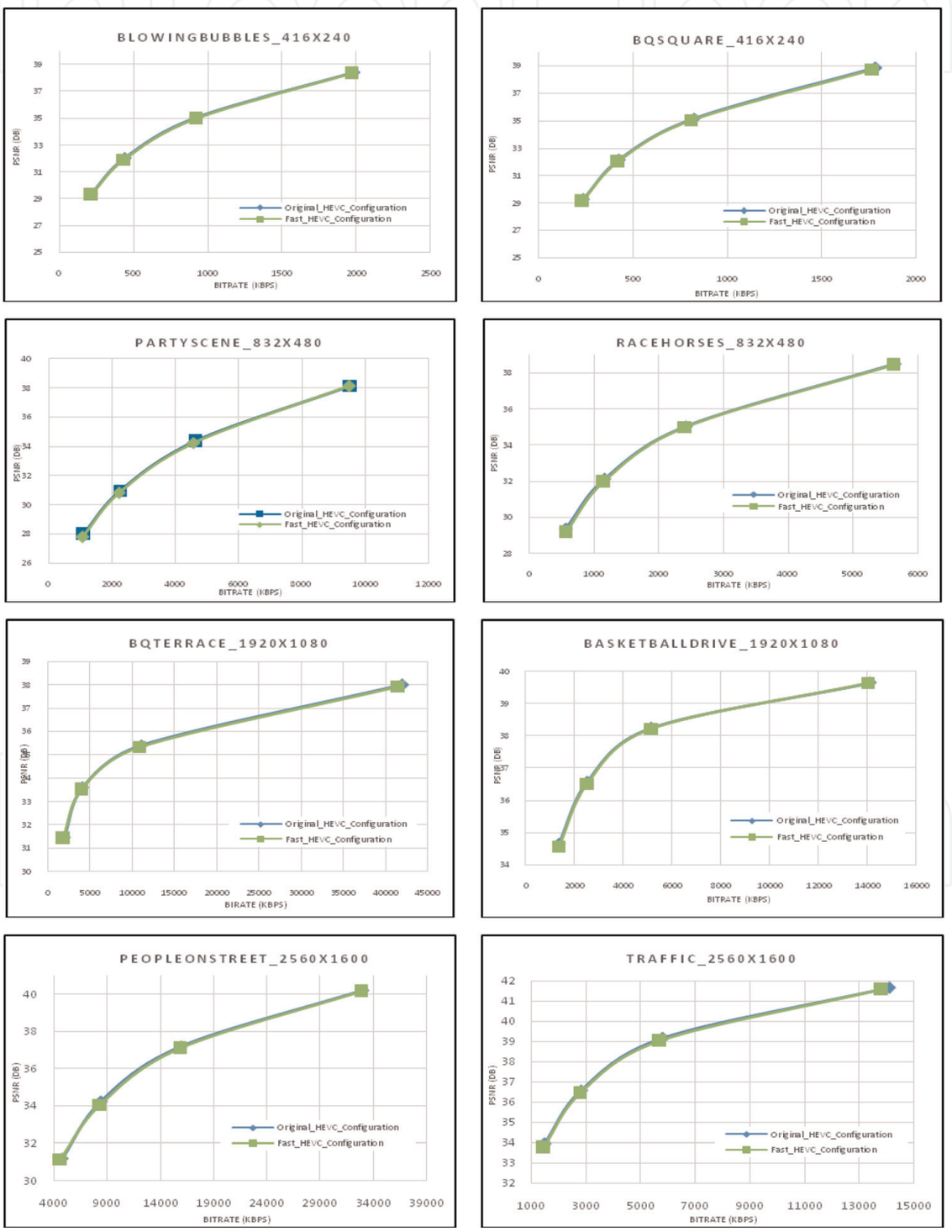

Figure 11.
RD curve comparison of our algorithm versus the original one.

racing. Many great frequency details are in this video, since horsetail is regularly expensive to encode.

The time saving is visible with 49% for BasketballDrive sequence. This video contains a high contrast and high motion activities. The background has a rather similar texture.

Not only the encoding time was saved but also the bitrate which is justified by the negative values in the table, ranging from 0.002 to −2.182% for PartyScene and PeopleOnStreet with QP equal to 22 and 37, respectively. Regarding the quality of video, the PSNR deprivation is from −0.015 to −0.23 dB for BasketballDrive and RaceHorses with QP equal to 22 and 37, respectively.

In average, the fast HEVC configuration induces a nonsignificant poverty in terms of video quality, around 0.106 dB, with a decrease of 0.416% in the bitrate that is a very interesting point in terms of increasing the compression performance.

Figure 11 shows the curves of rate distortion (RD) of HEVC original algorithm and the fast one, for two sequences for each class: PeopleOnStreet and Traffic from class A (2560 × 1600), BQTerrace and BasketballDrive from class B (1920 × 1080), PartyScene and BasketballDrive from class C (832 × 480), and BlowingBubbles and BQSquare from class D (416 × 240). This can also be checked in **Table 2**. The sequences are taken at QPs 22, 27, 32, and 37.

Four QP parameters are presented in all curves; horizontal axes on (kbps) represent the bitrate where the vertical one on (dB) represents the PSNR.

Figure 11 shows that all RD curves are overlaid [27]. In fact, the proposed changes have insignificant impairments on bitrate and PSNR. For lower QP values, the degradation is more significant. Experimental results prove that the fast configuration gives better performances than the original one, given that it offers a significant time saving, without any influence on the quality and the bitrate.

Further, for all tested sequences, an important speedup is obtained for bigger QPs. **Figure 12** evaluates the time saving in average by varying from 22 to 37. We note that the time saving increases in proportion to QP. In average, for higher QP, equal to 37, the run-time decreases by 63.5%. This decline is justified by the choice of the skip mode for bigger QP values [25].

Table 3 summarizes the performances of the proposed work compared to different previous algorithms.

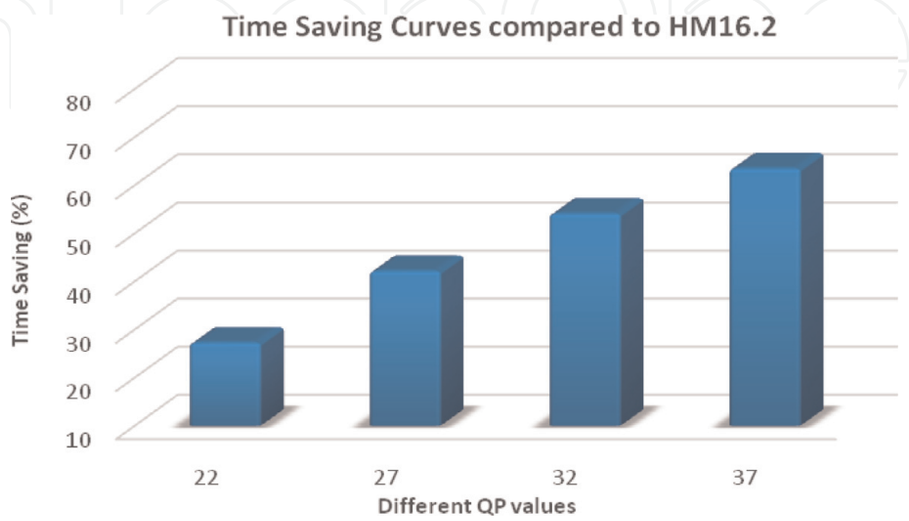

Figure 12.
Curves of time saving for all videos coded through random access configuration with QP from 22 to 37.

	Kibeya [17]			Liquan [19]			Qin [20]		
	ΔPSNR (dB)	ΔBR (%)	ΔT (%)	ΔPSNR (dB)	ΔBR (%)	ΔT (%)	ΔPSNR (dB)	ΔBR (%)	ΔT (%)
Class A	−0.052	1.08	30.67	—	—	—	0.1%	—	−22.4
Class B	−0.013	0.29	45.37	−0.020	0.834	−34.00	0.3%	—	−28.4
Class C	−0.011	0.53	20.86	−0.045	1.225	−16.50	0.2%	—	−23.0
Class D	−0.008	0.26	6.9	−0.040	1.060	−13. 50	0.2%	—	−17.0
Average	−0.0105	0.54	25.95	−0.035	1.039	−21.33	0.2%	—	−22.7
	Podder [21]			**Proposed fast algorithm**					
	ΔPSNR (dB)	ΔBR (%)	ΔT (%)	ΔPSNR (dB)	ΔBR (%)	ΔT (%)			
Class A	—	—	−41.9	−0.123	−0.478	−50.384			
Class B	—	—	−34.37	−0.059	−0.504	−55.926			
Class C	—	—	−42.92	−0.126	−0.454	−34.400			
Class D	—	—	−46.57	−0.116	−0.218	−46.326			
Average	—	—	−41.44	−0.106	−0.416	−46.759			

Table 3.
Proposed algorithm compared to previous works.

Compared to [17], the proposed work was more competent in terms of bitrate and saving time. In fact, [17] allows saving about 25.95% of encoding time with a slight bitrate. This algorithm was based on large diamond search pattern as an algorithm for motion estimation implemented on HM8.0. Concerning Liquan [19], its algorithm consists of skipping some detailed depths used in the preceding frames. This work allows saving about 21.5% of encoding time with a slight bitrate. Qin [20] implemented an algorithm established on the ECU according to a MSE adaptive threshold value. A time saving without degradation in the quality is obtained in this work. Another interesting method was presented by Podder et al. [21], where human visual features (HVF) are used for the selection of appropriate block partitioning modes. This work offered 41.44% reduction in terms of time for the standard class video sequences (SCVS).

6. Conclusion

HEVC induces an important progress in terms of video quality, in particular for high video resolutions. Nevertheless, this recital is combined with a bigger computational complexity which tremendously increases the encoding time. Motion estimation module using the quadtree structure represents the mainly strong process that is a conduit to the augmentation of the HEVC computational complication. In this paper to decrease this computational complexity, one fast configuration was presented to optimize the ME process by using CU partitioning fast mode decision algorithm and a diamond search. A reduction of 46.75% in the encoding time is obtained without inducing a significant degradation in encoding performance in terms of video quality or bitrate.

As perspectives, additional optimizations will be also implemented to reduce the encoder complexity via digital platform for video processing.

We will also exploit the fast configuration detailed in this paper for the new compression standard Joint Video Exploration Team (JVET) [28, 29].

IntechOpen

Author details

Randa Khemiri[1*], Nejmeddine Bahri[2], Fatma Belghith[2], Soulef Bouaafia[1], Fatma Elzahra Sayadi[1], Mohamed Atri[1] and Nouri Masmoudi[2]

1 Electronics and Microelectronics Laboratory, Monastir University, Faculty of Sciences of Monastir, Monastir, Tunisia

2 Laboratory of Electronics and Information Technology, National Engineering School of Sfax, Sfax University, Tunisia

*Address all correspondence to: randa.khemiri@gmail.com

IntechOpen

© 2019 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. CC BY

References

[1] ITU-T Recommendation H.265 and ISO/IEC 23008-2, ITU-T and ISO/IEC JTC 1. High Efficient Video Coding (HEVC). 2013. Available from: http://handle.itu.int/11.1002/1000/11885

[2] Sullivan GJ, Ohm J, Han WJ, Wiegand T. Overview of the high efficiency video coding (HEVC) standard. IEEE Transactions on Circuits and Systems for Video Technology. 2012;**22**(12):1649-1668

[3] Bahri N, Belhadj N, Grandpierre T, Ben Ayed MA, Masmoudi N, Akil M. Real-time H264/AVC encoder based on enhanced frame level parallelism for smart multicore DSP camera. Journal of Real-Time Image Processing. 2016; **12**(4):791-812

[4] Ohm J, Sullivan GJ, Schwarz H, Tan TK, Wiegand T. Comparison of the coding efficiency of video coding standards-including high efficiency video coding (HEVC). IEEE Transactions on Circuits and Systems for Video Technology. 2012;**22**(12): 1669-1684

[5] Kim J, Jun DS, Jeong S, Cho S, Choi JS, Kim J, et al. An SAD-based selective Bi-prediction method for fast motion estimation in high efficiency video CodingC. ETRI Journal. 2012;**34**(5): 753-758

[6] Yoo HM, Suh SJW. Fast coding unit decision algorithm based on inter and intra prediction unit termination for HEVC. In: IEEE International Conference on Consumer Electronics (ICCE). 2013. pp. 300-301

[7] Choi K, Park SH, Jang ES. Coding tree pruning based CU early termination, document JCTVC-F092. Joint Collaborative Team on Video Coding. 2011. Available from: https://www.itu.int/wftp3/av-arch/jctvc-site/2011_07_F_Torino/

[8] Yang J, Kim J, Won K, Lee H, Jeon B. Early SKIP detection for HEVC, document JCTVC-G543. Joint Collaborative Team on Video Coding. 2011. Available from: https://www.itu.int/wftp3/av-arch/jctvc-site/2011_07_F_Torino/

[9] Bossen F, Flynn D, Sharman K, Suhring K. Joint collaborative team on video coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11. HM Software Manual; 2014. Available from: https://usermanual.wiki/Pdf/softwaremanual.2114962307/help

[10] Software reference test model HM16.2. 2016. Available from: https://hevc.hhi.fraunhofer.de/

[11] Khemiri R, Kibeya H, Loukil H, Sayadi FE, Atri M, Masmoudi N. Real-time motion estimation diamond search algorithm for the new high efficiency video coding on FPGA. Analog Integrated Circuits and Signal Processing. 2018;**94**(2):259-276. DOI: https://doi.org/10.1007/s10470-017-1072-6

[12] Rehman S, Young R, Chatwin C, Birch P. An FPGA based generic framework for high speed sum of absolute difference implementation. European Journal of Scientific Research. 2009;**33**(1):6-29

[13] Nalluri P, Alves LN, Navarro A. A novel SAD architecture for variable block size motion estimation in HEVC video coding. In: IEEE International Symposium on System on Chip (SoC). 2013. pp. 1-4

[14] Nalluri P, Alves LN, Navarro A. High speed SAD architectures for variable block size motion estimation in HEVC video coding. In: IEEE International Conference on Image Processing (ICIP). 2014. pp. 1233-1237

[15] Khemiri R, Kibeya H, Sayadi FE, Bahri N, Atri M, Masmoudi N. Optimisation of HEVC motion estimation exploiting SAD and SSD GPU-based implementation. IET Image Processing. 2018;**12**(2):243-253

[16] Nalluri P, Alves LN, Navarro A. Improvements to TZ search motion estimation algorithm for multiview video coding. In: International Conference on Systems, Signals and Image Processing (IWSSIP); Austria. 2015. pp. 388-301

[17] Kibeya H, Belghith F, Ben Ayed MA, Masmoudi N. Fast coding unit selection and motion estimation algorithm based on early detection of zero block quantified transform coefficients for high-efficiency video coding standard. IET Image Processing. 2016;**10**(5): 371-380

[18] Belghith F, Kibeya H, Loukil H, Ben Ayed MA, Masmoudi N. A new fast motion estimation algorithm using fast mode decision for high-efficiency video coding standard. Journal of Real-Time Image Processing. 2014;**11**(4):675-691

[19] Liquan S, Zhi L, Xinpeng Z, Wenqiang Z, Zhaoyang Z. An effective CU size decision method for HEVC encoders. IEEE Transactions on Multimedia. 2013;**15**(2):465-470

[20] Qin Y, Xinfeng Z, Siwei M. Early termination of coding unit splitting for HEVC. In: IEEE Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC); Asia-Pacific. 2012

[21] Podder PK, Paul M, Murshed M. Fast mode decision in the HEVC video coding standard by exploiting region with dominated motion and saliency features. PLoS One. 2016;**11**(3):1-12. Available from: https://doi.org/10.1371/journal.pone.0150673

[22] Khemiri R, Bahri N, Belghith F, Sayedi FE, Atri M, Masmoudi N. Fast motion estimation for HEVC video coding. In: IEEEIPAS'16: International Image Processing Applications and Systems Conference. 2016. pp. 1-4

[23] Gweon RH, Lee JL, Lim J. Early termination of CU encoding to reduce complexity HEVC. JCTVC-F045. 2011. Available from: https://www.itu.int/wftp3/av-arch/jctvc-site/2011_07_F_Torino/

[24] Liu H, Jie Y. Fast HEVC intra mode decision based on hybrid cost ranking. Advances in Multimedia. 2016;**3**:1-9. London: Hindawi Publishing Corporation; DOI: http://dx.doi.org/10.1155/2016/5164025

[25] Frank B, Benjamin B, Karsten S, David F. HEVC complexity and implementation analysis. IEEE Transactions on Circuits and Systems for Video Technology. 2012;**22**((12): 1685-1696

[26] Dıaz-Honrubia A, Martinez J, Cuenca P. HEVC: A Review, Trends and Challenges. Second Workshop on Multimedia Data Coding and Transmission. 2011

[27] Nalluri P, Alves LN, Navarro A. Fast motion estimation algorithm for HEVC. In: IEEE Second International Conference on Consumer Electronics. 2012

[28] ISO/IEC. ITU-T Joint Exploration Test Model (JEM) Reference Software. 2017. Available from: https://jvet.hhi.fraunhofer.de/

[29] García-Lucas D, Cebrián-Márquez G, Díaz-Honrubia AJ, Cuenca P. Acceleration of the integer motion estimation in JEM through pre-analysis techniques. Journal of Supercomputing. March 2019;**75**(3):1203-1214. DOI: 10.1007/s11227-018-2352-3